APPLE 1
QUICK REFERENCE

Version 0.97

MONITOR COMMANDS

DUMP MEMORY EXAMPLES

Contents of 4F

```
INPUT:    4F(RET)
OUTPUT:   004F: 0F
```

From last location displayed to higher location

```
INPUT:    .5A(RET)
OUTPUT:   0050: 00 01 02 03 04 05 06 07
          0058: 08 09 0A
```

Note: 4F is still considered the most recently opened location

Specify start and end

```
INPUT:    4F.5A (RET)
OUTPUT:   004F: 0F
          0050: 00 01 0a 03 04 05 06 07
          0058: 08 09 0A
```

Dump several locations or blocks

```
INPUT:    4F 52 56 58.5A(RET)
OUTPUT:   004F: 0F
          0052: 02
          0056: 06
          0058: 08 09 0A
```

STORE INTO MEMORY

Store data into a single location.

```
INPUT:    30:A1 (RET)
OUTPUT:   30: FF
```

Note: Prior contents displayed

Depositing data in successive locations from the last address used in a store command

```
INPUT:    :A1 A2 A3 A4 A5(RET)
```

Note: This deposits A1 in location 31, A2 in 32, and so on

RUNNING A PROGRAM

Run a program at address E000. This example initializes and starts BASIC, if it is loaded.

```
INPUT: E000R(RET)
OUTPUT: E000: 4C
```

Note: Contents of E000 are displayed

Run a program at address E2B3. This example restarts BASIC, without clearing any loaded BASIC program.

```
INPUT: E2B3R(RET)
OUTPUT: E2B3: 20
```

Note: Contents of E2B3 are displayed

SIMPLE TERMINAL OUPUT EXERCISE

1: Press the clear key

2: Press the reset key

3: You should see a backslash "\"prompt

4: Type the following command

5: 0:A9 0 AA 20 EF FF E8 8A 4C 2 0(RET)

6: 0R(RET) to run the program (0 = zero)

7: Press the reset key to stop the program

IMPORTANT ADDRESSES

ADDRESS	DESCRIPTION
0-FFF	First bank of DRAM (4K)
0-3FFF	First bank of DRAM (16K mod)
0-FF	Page zero
24-2E	Used as index pointers by the monitor
100-1FF	Stack - stack grows from top down
200-27F	Monitor keyboard input buffer
C000	Write -Toggle cassette output
C000	Read - Cassette read input register data toggles every transition
C100-C1FF	Cassette interface driver
D000-D01F	6820 PIA registers
D010	Read keyboard data
D011	Read - MSB bit is one when keyboard data is present
D012	Read - MSB is one when termi nal is ready for data
D012	Write - write byte to terminal
E000-EFFF	Second Bank of DRAM (4K) - BASIC is normally loaded here
E000	BASIC initalization entry
E2B3	BASIC restart entry
FF00-FFFF	256 byte PROM monitor
FFEF	Monitor putchar routine

BASE CONVERSIONS

HEX:	1	2	4	8	10	20	40	80
DECIMAL:	1	2	4	8	16	32	64	128

APPLE CASSETTE INTERFACE (ACI)

ENTERING AND EXITING ACI DRIVER

INPUT: C100R(RET)
OUTPUT: C100: A9

ACI returns to monitor upon command completion or press reset. Always enter leading zeros when entering an address. Stop tape after reading or writing.

READ EXAMPLE: LOADING AND RUNNING BASIC

If the tape volume is set correctly, this will load BASIC into memory locations E000 through EFFF. Enter the ACI driver and position the tape at the start of BASIC. 'R' denotes read

INPUT: E000.EFFFR

Start the cassette player

INPUT: (RET)

After the read, the computer will return to monitor. Check for a good read by examining the last two bytes and verifying that they contain 0C and E0

INPUT: EFFE.EFFF(RET)
OUPUT: EFFE: 0C E0

To delete any existing program and variables and run BASIC

INPUT: E000R(RET)

READ EXAMPLE: LOAD BASIC PROGRAM (HAMMURABI)

Position tape at beginning of Hammurabi and enter ACI driver using C100R (ret)

INPUT: 004A.00FFR 0400.0FFFR

Start the tape

INPUT: (RET)

this should load the BASIC variables needed to run the program into memory locations 004A.00FF and the BASIC program iself into memory 0400.0FFF

After the computer returns to the monitor, check for a good load by reading FFE.FFF and verifying it contains 51 01

INPUT: FFE.FFF(RET)
OUPUT: 0FFE: 51 01

Return control of the processor to BASIC without deleting the loaded program

INPUT: E2B3R(RET)

Run HAMMURABI

INPUT: RUN(RET)

WRITE EXAMPLE: Save BASIC PROGRAM (HAMMURABI)

Write is the same as read, except that a W is used instead of R.

INPUT: 004A.00FFW 0400.0FFFW

Start tape in record mode

INPUT: (RET)

6502 QUICK REFERENCE

REGISTERS

A	Accumulator
X	index register
Y	index register
PC	program counter (high and low bytes)
S	stack pointer
P	status register flags

ADDRESSING MODES

-	implied	1 byte	variable cycles
A	accumulator	1 byte	2 cycles
I	immediate	2 bytes	2 cycles
Z	zero page	2 bytes	3 cycles
ZX	zero page,x	2 bytes	4 cycles
ZY	zero page,y	2 bytes	4 cycles
AB	absolute	3 bytes	4 cycles
(AB)	abs,indirect	3 bytes	5 cycles
AX	absolute,x	3 bytes	4 cycles[1]
AY	absolute,y	3 bytes	4 cycles[1]
(IX)	(indirect,x)	2 bytes	6 cycles
(I)Y	(indirect),Y	2 bytes	5 cycles[1]
R	relative	2 bytes	2/3/4 cycles

STATUS REGISTER FLAGS

N	V		B	D	I	Z	C
0x80	0x40		0x10	0x08	0x04	0x02	0x01

N	NEGATIVE	D	DECIMAL MODE
V	OVERFLOW	I	INTERUPT DISABLE
B	BREAK INST.	Z	ZERO
		C	CARRY

HISTORIC
COMPUTERS
REPRODUCTIONS
HARDWARE
SOFTWARE

EMAIL:mike@willegal.net

INSTRUCTION SET

BRANCH (FLAGS NOT AFFECTED, RELATIVE ADDRESSING)

Op	Hex	Description	Cycles
BCC	90	Branch if C=0	2/3/4
BCS	B0	Branch if C=1	2/3/4
BEQ	F0	Branch if Z=1	2/3/4
BNE	D0	Branch if Z=0	2/3/4
BMI	30	Branch if N=1	2/3/4
BPL	10	Branch if N=0 2 no branch	2/3/4
BVC	50	Branch if V=0 3 same page	2/3/4
BVS	70	Branch if V=1 4 diff. page	2/3/4

JUMP, CALL AND RETURN (ONLY RTI AFFECTS FLAGS)

Op	Hex	Description	Cycles
JMP	4C	AB ▸ PC	3
JMP	6C	(AB) ▸ PC	5
JSR	20	PC+2 ▸ (S-2), AB ▸ PC	6
BRK	00	PC+2▸(S-2),P▸(S--),FFFE▸PC,1▸B	7
RTS	60	(2+S) ▸ PC	6
RTI	40	(++S) ▸ P, (2+S) ▸ PC	6

FLAG MANIPULATION (IMPLIED ADDRESSING)

Op	Hex	Description	Cycles	NVBDIZC
CLC	18	0 ▸ C	2	------0
CLD	D8	0 ▸ D	2	---0---
CLI	58	0 ▸ I	2	----0--
CLV	B8	0 ▸ V	2	-0-----
SEC	38	1 ▸ C	2	------1
SED	F8	1 ▸ D	2	---1---
SEI	78	1 ▸ I	2	----1--

NOP AND REGISTER MOVE (IMPLIED ADDRESSING)

Op	Hex	Description	Cycles	NVBDIZC
NOP	EA	No Operation	2	-------
TAX	AA	A ▸ X	2	√----√-
TXA	8A	X ▸ A	2	√----√-
TAY	A8	A ▸ Y	2	√----√-
TYA	98	Y ▸ A	2	√----√-
TSX	BA	S ▸ X	2	√----√-
TXS	9A	X ▸ S	2	-------
PHA	48	A ▸ (S--)	3	-------
PLA	68	(++S) ▸ A	4	√----√-
PHP	08	P ▸ (S--)	3	-------
PLP	28	(++S) ▸ P	4	√√-√√√√

MATH AND COMPARE

Op	Description	Flags
ADC	**A + M + C ▸ A.C**	

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
69	65	75	6D	7D	79	61	71	√√---√√

SBC **A - M - $\overline{C}$ ▸ A.C**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
E9	E5	F5	ED	FD	F9	E1	F1	√√---√√

CMP **A - M**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
C9	C5	D5	CD	DD	D9	C1	D1	√----√√

CPX **X - M**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
E0	E4	--	EC	--	--	--	--	√----√√

CPY **Y - M**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
C0	C4	--	CC	--	--	--	--	√----√√

BOOLEAN

AND **A & M ▸ A**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
29	25	35	2D	3D	39	21	31	√----√-

ORA **A | M ▸ A**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
09	05	15	0D	1D	19	01	11	√----√-

EOR **A ^ M ▸ A**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
49	45	55	4D	5D	59	41	51	√----√-

BIT **A & M ▸ Z M7->N M6->V**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
--	24	--	2C	--	--	--	----	76---√-

LOAD AND STORE

LDA **M ▸ A**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
A9	A5	B5	AD	BD	B9	A1	B1	√----√-

LDX **M ▸ X**

I	Z	ZY	AB	AX	AY	(IX)	(I)Y	NVBDIZC
A2	A6	B6	AE	--	BE	--	--	√----√-

LDY **M ▸ Y**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
A0	A4	B4	AC	BC[1]	--	--	--	√----√-

STA **A ▸ M**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
--	85	95	8D	9D[5]	99[5]	81	91[6]	-------

STX **X ▸ M**

I	Z	ZY	AB	AX	AY	(IX)	(I)Y	NVBDIZC
--	86	96	8E	--	--	--	--	-------

STY **Y ▸ M**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
--	84	94	8C	--	--	--	--	-------

ROTATE

ASL **M << 1, MSB ▸ C, 0 ▸ LSB**

A	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
0A	06[5]	16[6]	0E[6]	1E[7]	--	--	--	√----√√

LSR **M >> 1, LSB ▸ C, 0 ▸ MSB**

A	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
4A	46[5]	56[6]	4E[6]	5E[7]	--	--	--	0----√√

ROL **M << 1, MSB ▸ LSB, MSB ▸ C**

A	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
2A	26[5]	36[6]	2E[6]	3E[7]	--	--	--	√----√√

ROR **M >> 1, LSB ▸ MSB, LSB ▸ C**

A	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
6A	66[5]	76[6]	6E[6]	7E[7]	--	--	--	√----√√

INCREMENT/DECREMENT

DEC **M - 1 ▸ M**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
--	C6[5]	D6[6]	CE[6]	DE[7]	--	--	--	√----√-

INC **M + 1 ▸ M**

I	Z	ZX	AB	AX	AY	(IX)	(I)Y	NVBDIZC
--	E6[5]	F6[6]	EE[6]	FE[7]	--	--	--	√----√-

DEX	CA	X - 1 ▸ X	2 cycles	√----√-
DEY	88	Y - 1 ▸ Y	2 cycles	√----√-
INX	E8	X + 1 ▸ X	2 cycles	√----√-
INY	C8	Y + 1 ▸ Y	2 cycles	√----√-

[1]add 1 cycle if page crossed [6]6 cycles
[5]5 cycles [7]7 cycles